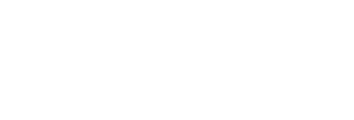
#Shruti Baravkar

#121B1B067

Assignment No. 6



Write a program to simulate memory allocation techniques: First Fit, Best Fit, Next Fit and Worst Fit. Code :

#include <stdio.h>

void firstFit(int blockSize[], int m, int processSize[], int n)

{

int i, j;

int allocation[n];

for(i = 0; i < n; i++) { allocation[i] = -1;

}

for (i = 0; i < n; i++) { for (j = 0; j < m; j++) {

if (blockSize[j] >= processSize[i])

{

allocation[i] = j;

blockSize[j] -= processSize[i]; break;

} } }

printf("\nFirst Fit\n");

printf(" \n");

printf("\nProcess\tProcess Size\tBlock no.\n"); for (int i = 0; i < n; i++){

printf("P[%d] \t\t %d \t", i+1, processSize[i]); if (allocation[i] != -1)

printf("%d\n",allocation[i] + 1); else

printf("Not Allocated\n");

}

}

void bestFit(int blockSize[], int blocks, int processSize[], int processes){ int allocation[processes];

for(int i = 0; i < processes; i++){ allocation[i] = -1;

}

for (int i=0; i < processes; i++) { int indexPlaced = -1;

for (int j=0; j < blocks; j++) {

if (blockSize[j] >= processSize[i]) { if (indexPlaced == -1)

indexPlaced = j;

else if (blockSize[j] < blockSize[indexPlaced]) indexPlaced = j;

}}

if (indexPlaced != -1){ allocation[i] = indexPlaced;

blockSize[indexPlaced] -= processSize[i];}}

printf("\nBest Fit\n");

printf(" \n");

printf("\nProcess\tProcess Size\tBlock no.\n"); for (int i = 0; i < processes; i++) {

printf("P[%d] \t\t %d \t", i+1, processSize[i]); if (allocation[i] != -1)

printf("%d\n",allocation[i] + 1); else

printf("Not Allocated\n"); }

}

void worstFit(int blockSize[], int blocks, int processSize[], int processes){

int allocation[processes];

for(int i = 0; i < processes; i++){ allocation[i] = -1;

}

for (int i=0; i<processes; i++){ int indexPlaced = -1;

for (int j=0; j<blocks; j++)

{

if (blockSize[j] >= processSize[i]) { if (indexPlaced == -1)

indexPlaced = j;

else if (blockSize[indexPlaced] < blockSize[j]) indexPlaced = j;}}

if (indexPlaced != -1) { allocation[i] = indexPlaced;

blockSize[indexPlaced] -= processSize[i];

}

}

printf("\nWorst Fit\n");

printf(" \n");

printf("\nProcess\tProcess Size\tBlock no.\n"); for (int i = 0; i < processes; i++) {

printf("P[%d] \t\t %d \t", i+1, processSize[i]); if (allocation[i] != -1)

printf("%d\n",allocation[i] + 1); else

printf("Not Allocated\n");

}

}

void nextFit(int blockSize[],int m,int processSize[],int n){ int allocation[n];

for (int i=0;i<n;i++){ allocation[i]=-1;

}

int j=0;

for (int i = 0; i < n; i++) { int count=0;

while (count < m) {

if (blockSize[j] >= processSize[i]) { allocation[i] = j;

blockSize[j] -= processSize[i]; break;

}

j = (j + 1) % m; count+=1;

}

}

printf("\nNext Fit\n");

printf(" \n");

printf("\nProcess\tProcess Size\tBlock no.\n"); for (int i = 0; i < n; i++) {

printf("P[%d] \t\t %d \t", i+1, processSize[i]); if (allocation[i] != -1)

printf("%d\n",allocation[i] + 1); else

printf("Not Allocated\n");

}

}

int main()

{ int blocks, processes;

printf("Enter number of blocks : ");

scanf("%d", &blocks);

printf("Enter number of processes : "); scanf("%d", &processes);

int blockSize[blocks];

int processSize[processes];

printf("Enter block sizes : "); for (int i = 0; i < blocks; i++){

scanf("%d", &blockSize[i]);

}

printf("Enter process sizes : "); for (int i = 0; i < processes; i++){

scanf("%d", &processSize[i]);

}

int blockSizeCopy1[blocks]; for (int i = 0; i < blocks; i++)

blockSizeCopy1[i] = blockSize[i];

firstFit(blockSizeCopy1, blocks, processSize, processes);

int blockSizeCopy2[blocks]; for (int i = 0; i < blocks; i++)

blockSizeCopy2[i] = blockSize[i];

bestFit(blockSizeCopy2, blocks, processSize, processes);

int blockSizeCopy3[blocks]; for (int i = 0; i < blocks; i++)

blockSizeCopy3[i] = blockSize[i];

worstFit(blockSizeCopy3, blocks, processSize, processes);

int blockSizeCopy4[blocks]; for (int i = 0; i < blocks; i++)

blockSizeCopy4[i] = blockSize[i];

nextFit(blockSizeCopy4, blocks, processSize, processes);

return 0 ;

}

Output :

